Real-Time Emotion Recognition for Companion Robots Using Deep Learning and Edge Computing

1st Yahia Salman
Computer Science Department
George Mason University
Fairfax, USA
ysalman2@gmu.edu

2nd Muhammad Irfan
Computer Science Department
James Madison University
Harrisonburg, USA
irfan2mx@dukes.jmu.edu

3rd Mohammad Shike
Computer Science Department
James Madison University
Harrisonburg, USA
shikemi@dukes.jmu.edu

4th Suk Jin Lee
Computer Science Department
James Madison University
Harrisonburg, USA
lee43sj@jmu.edu

5th Ahmad Salman
Computer Science Department
James Madison University
Harrisonburg, USA
salmanaa@jmu.edu

***Abstract*—With growing public health concerns related to conditions such as anxiety and depression, the need for modern technology to support affected individuals has become increasingly important. This paper details the development of a companion robot that leverages edge computing for real-time emotion recognition. Our approach utilizes a Convolutional Neural Network (CNN), fine-tuned from the ResNet-50 base model, to identify five distinct emotions: Happy, Neutral, Sad, Angry, and Fear. Video frames are captured using a Raspberry Pi camera and sent to the edge device, where face detection and emotion inference are performed. Once the inference and analysis are complete, the results are transmitted back to the Raspberry Pi. Device communication is facilitated through the HTTP protocol. Our model achieved a validation accuracy of 80% while maintaining an average communication time of 76.1 ms, including the duration for model inference.**

***Index Terms*—Artificial Intelligence, Machine Learning, Edge Computing, Robotics, Secure Communications**

I. Introduction

Emotion detection from facial expressions is a rapidly advancing field within computer vision [1]. Research in this area has proven to be pivotal in developing solutions that range from diagnosing emotion-related diseases in healthcare settings to assisting with rehabilitation in home environments [2].

Recent studies highlight the immense potential of emotion recognition technology, particularly in healthcare applications [3]. This technology can be instrumental in monitoring mental health conditions such as anxiety and depression, which have become even more critical since the COVID-19 pandemic [3]. The pandemic not only heightened global awareness of mental health issues but also spurred a search for technological solutions to mitigate the rise in emotional distress. By enabling early detection of emotional distress through advanced Artificial Intelligence (AI) systems, early interventions can be facilitated, potentially improving outcomes and saving lives.

The integration of emotion recognition technology into real-time applications presents a significant opportunity to enhance quality of life, particularly for individuals experiencing loneliness or emotional distress. Companion robots equipped with emotion recognition capabilities can offer substantial support in healthcare, elderly care, and even child care. The prospect of developing emotionally intelligent robots capable of responding empathetically to human emotions is a major driving force behind this research.

Our project aims to advance the current state of technology by developing a companion robot that utilizes the computational power of portable edge devices (such as laptops or mobile embedded systems) to recognize human emotions in real-time. Achieving this requires a robust communication architecture and a facial emotion recognition model optimized for resource-constrained hardware environments. This paper focuses on the design of the communication architecture and the development of a facial emotion recognition model suited for deployment on edge devices.

The rest of this paper is structured as follows: Section II discusses related work in the fields of intelligent companion robots and edge computing. Section III details the system design and its components. Section V presents our results and their analysis. Section VI addresses open challenges, and Section VII concludes the paper.

II. Related Work

A key component of this project is the use of the Histogram of Oriented Gradients (HOG) algorithm for both face detection and emotion classification. Dalal et al. [4] explored HOG for human detection, and a similar approach was applied here to detect faces and extract features for emotion recognition.

Boer et al. [5] compared various pre-trained Deep Convolutional Neural Networks for facial emotion detection, identifying VGG19 as the most effective. However, in our experiments, ResNet-50 outperformed VGG19, achieving a validation accuracy of 80%, compared to VGG19's 64.89%.

Gong [6] argued that transfer learning may not generalize well for emotion recognition. Although their implementation of ResNet-50 resulted in lower accuracy post-transfer learning, we fine-tuned ResNet-50 and achieved significantly better performance, with a validation accuracy of 80%.

Chowdary et al. [7] reported a 97% accuracy using a fine-tuned ResNet-50 model on the CK+ dataset. In contrast, our approach incorporates ResNet-50 within a companion robot-edge system, requiring faster inference times, which led to a slight decrease in accuracy. We used the more diverse RAF-DB [8] dataset, while Chowdary's study was based on the CK+ dataset with a limited number of subjects.

Nimmagadda et al. [9] presented a humanoid robot for emotion detection using facial and audio data. Their model was trained on the FER-13 dataset, whereas we employed the RAF-DB dataset for better image diversity. Additionally, while their computation was done on the robot, our system leverages an edge device to optimize performance on lower-end hardware like the Raspberry Pi 4 Model B+.

We propose a companion robot-edge device system for emotion recognition using a fine-tuned ResNet-50 model, where model computation occurs on the edge device. This allows resource-constrained devices like the Raspberry Pi to benefit from GPU-level performance without requiring such hardware.

III. System Design

A. Communication Architecture

The system adopts a client-server architecture where the Raspberry Pi 4 Model B+ acts as the client, capturing video frames and sending them to an edge device for processing. The Raspberry Pi receives computed data back from the edge device to activate actuators. Due to the Pi's hardware limitations, the more intensive computations, such as running machine learning models, are performed on the edge device.

Video frames are captured via a 5MP Raspberry Pi camera, converted into pixel arrays, and sent to the edge device in JSON format via a POST request. The edge device processes the data and sends the detected emotion back to the Pi, which then triggers appropriate responses. Communication between devices is secured using HTTPS to protect user privacy, ensuring that video data remains secure within the local network.The communication architecture can be seen in Fig. 1

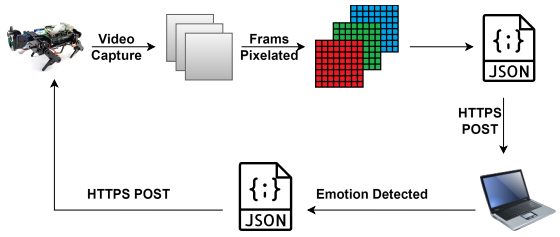


Fig. 1: Communication Architecture

B. Artificial Intelligence Model

The core AI model is based on a Convolutional Neural Network (CNN) architecture, specifically ResNet-50, which was optimized for emotion classification using transfer learning.

1) Model Architecture: The model is based on the ResNet-50 architecture, a 50-layer Convolutional Neural Network (CNN) pre-trained on the ImageNet dataset and is highly optimized for computer vision tasks like image classification and object detection. We applied transfer learning to leverage ResNet-50's feature extraction capabilities, while fine-tuning additional fully connected layers for our specific emotion classification task. The input to the model is a resized 224x224 image with 3 color channels (RGB), which conforms to ResNet-50's input requirements.

ResNet-50's pre-trained layers perform feature extraction, producing a 7x7x2048 feature map from the input image. The output size of the image after each convution layer is calculated using Eq. 1. where o represents the output size of the feature map, i is the input dimensions of the image that is going to processed, p is the padding on the side of the image, k is the kernel size and s is the stride.

$$o = \left\lfloor \frac{i + 2p - k}{s} \right\rfloor + 1 \text{ [10]} \tag{1}$$

During feature extraction, the weights of ResNet-50 are frozen to maintain the learned representations from ImageNet. The feature map is flattened into a 1D vector with 100,352 elements, which is passed to a custom fully connected network. This network consists of two dense layers with 512 and 128 neurons, respectively. To reduce overfitting, the second dense layer includes L1-L2 regularization [11], a method that drives weights of less important features towards 0. Additionally, Both layers employ the Rectified Linear Unit (ReLU) activation function (Eq. 2) to introduce non-linearity enabling the model to capture complex patterns. The term n can be represented by Eq. 3, where **W** is the weight matrix, **a** is the output vector from the previous layer and **b** is the bias vector.

$$f(n) = max(0, n) \tag{2}$$

$$n = \mathbf{W}\mathbf{a} + \mathbf{b} \tag{3}$$

Each dense layer is followed by a dropout layer with a 30% dropout rate to further mitigate overfitting. The final output layer consists of units corresponding to the predicted emotion classes. The softmax function is used to convert the outputs into probability distributions, facilitating multi-class classification.

The flow of the model, from face detection to emotion classification, is illustrated in Fig. 2

2) Facial Detection: Before passing through the model, each frame captured by the 5MP Raspberry Pi camera is analyzed to detect faces. This is done using the Histogram of Oriented Gradients (HOG) [4] and a Linear Support Vector

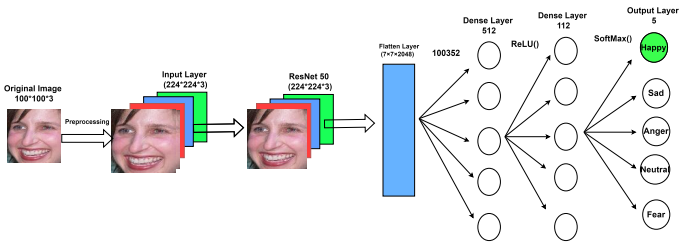


Fig. 2: Model Architecture

Machine (SVM). To simplify the gradient computation for each pixel, the image is first converted to grayscale by calculating the weighted sums of the RGB values, as shown in Eq 4.

$$I_{gray} = 0.299 * R + 0.587 * G + 0.114 * B \text{ [12]} \quad (4)$$

Gradients are then computed for each pixel to capture edge information. The gradients in the x and y directions (G_x and G_y) are calculated using simple derivative masks as shown in Eq. 5 and Eq. 6:

$$G_x = I(x+1, y) - I(x-1, y) \text{ [4]} \quad (5)$$

$$G_y = I(x, y+1) - I(x, y-1) \text{ [4]} \quad (6)$$

The gradient's magnitude and orientation are then computed using Eq. 7 and Eq. 8:

$$\text{Magnitude}(x, y) = \sqrt{G_x(x, y)^2 + G_y(x, y)^2} \text{ [4]} \quad (7)$$

$$\theta(x, y) = \arctan\left(\frac{G_y(x, y)}{G_x(x, y)}\right) \text{ [4]} \quad (8)$$

These gradients are used to create histograms of oriented gradients, capturing the gradient direction distribution. A histogram function H for each bin i is shown in Eq. 9

$$H(i) = f_i \text{ [4]} \quad (9)$$

Here, f_i represents the weighted sum of gradient magnitudes for orientations within that bin. The histograms are normalized and provided as input features to the Linear SVM for classification.

3) Emotion Recognition: When an image is processed by the ResNet-50 model, HOG is used to detect facial landmarks. These features, combined with convolutions operators, which are the kernels used in ResNet 50 to extract features, form the feature vector during the pre-trained stage. The outputted feature vector is then passed through the custom fully connected network architecture described in Section III-B1. The final output layer applies the softmax function shown in Eq. 10 where i is the index of the emotion in the output vector, K is the number of emotions in the output vector, and z is the corresponding output at the i'th location.

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad (10)$$

This converts the output into a vector of probabilities between 0 and 1 for each emotion, where the emotion with the highest probability is selected by the companion robot.

C. System Training

We utilized the RAF-DB dataset to train all models. Before training, images were resized to 224x224 pixels to match the ResNet-50 input requirements. Additionally, random image augmentations, such as flipping, rotation, zoom, and contrast adjustments, were applied to reduce overfitting.

A total of 8 models were trained. The primary model classified five emotions: Happy, Sad, Neutral, Fear, and Anger.

The remaining models were trained similarly but used subsets of emotions. As discussed in Section VI, ambiguity was especially common between emotions like neutral and sad or anger and surprise, likely due to the subtle differences in facial expressions. For instance, sadness may not always involve a frown but rather an unbothered expression, making the nuances difficult for the model to distinguish.

IV. Evaluation Metrics

The following performance metrics were used to evaluate the efficiency and reliability of our model and communication architecture.

A. Communication Latency

This is the time taken by the robot (Raspberry Pi) to send the pixel arrays to the edge device through a post request using the HTTP protocol and the edge device to send back the possible prediction to the Raspberry Pi using another post request in the form of a JSON file.

Inference Time: This is the time taken by the edge device to predict emotion.

Total Time: This is the total amount of time taken by the architecture from capturing the frame to predicting an emotion shown in Eq. 11.

$$\text{Total Time} = \text{Communication Time} + \text{Inference Time} \quad (11)$$

B. Loss

Training loss measures how well the model performs on seen data after each epoch. It is calculated based on the errors made during predictions on training data. The lower the training loss, the fewer errors the model made. We used Categorical Cross-Entropy as the loss function, assigning a value of 1 to the true class and 0 to others as shown in Eq. 12.

$$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(p_{ij}) \quad (12)$$

Where N is the number of samples, C is the number of classes, y_{ij} is the binary indicator for class label j of sample i, and p_{ij} is the predicted probability for class label j of sample i.

Validation loss measures model performance on unseen data after each epoch, indicating how well the model generalizes. Similar to training loss, we used Categorical Cross-Entropy for validation, calculated similarly as shown in Eq. 12

C. Accuracy

Training accuracy measures the number of correct predictions on seen data after each epoch. Higher training accuracy means more correct predictions. We used categorical accuracy, calculated by dividing the number of correct predictions by the total number of predictions, as shown in Eq. 13

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \tag{13}$$

Validation accuracy measures the model's performance on unseen data after each epoch, representing how well it generalizes. The validation accuracy is calculated by analyzing the number of correct predictions a model made while performing predictions on unseen data. The higher the validation accuracy, the higher the number of correct predictions. The accuracy metric that we used to assess validation in our model is the standard categorical accuracy, where a vector value of 1 is assigned to the true class and 0 to all others.

For multi-class classification problems, categorical accuracy calculates the ratio of correctly predicted samples to the total number of samples as shown in Eq. 13:

V. Results and Analysis

A. M2M Communication

Communication between the Raspberry Pi 4 and the edge device is facilitated via the HTTPS protocol, with the edge device powered by an AMD Ryzen 5 5600X CPU and utilizing an NVIDIA RTX 2070 Super GPU for model inference.

As depicted in Fig. 3, the majority of the time is consumed by the inference process.

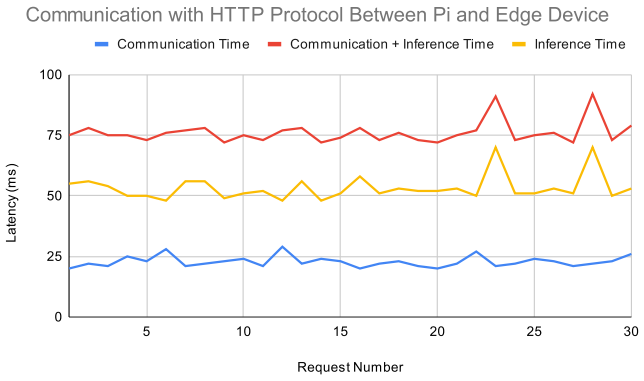


Fig. 3: Communication Latency Between Pi and Edge Device

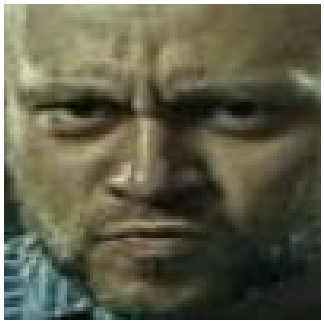
The average time for the entire communication process, including emotion inference, is 76.1 ms, with a median of 75 ms. This results in an effective frame rate of 13.14 FPS for the robot dog. While the inference time could be improved with more powerful hardware, such as the NVIDIA A100 in the Jetson Orin Nano, the communication time itself is largely unaffected by hardware changes and is primarily influenced by the network speed between the robot and the edge device.

To enhance overall performance, asynchronous programming was implemented. If the Raspberry Pi received an inference identical to the previous one, it would ignore the new inference, preventing the robot dog from repeating the same action unnecessarily. Additionally, the edge device did not wait for the robot to complete its movement before beginning the next inference.

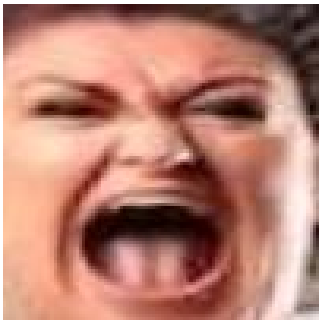
To mitigate the effect of incorrect inferences, the robot only executed a motion when the current inference matched the majority of the last 60 inferences.

B. Performance Evaluation

As discussed in Section III-C, eight models were trained, with four focusing on more extreme versions of negative emotions to address confusion in distinguishing between them. While accuracy improved across all models, both training and validation loss increased significantly, indicating greater overfitting. This can be attributed to the removal of more nuanced versions of emotions. When presented with a different type of extreme emotion within the same category during testing, the model struggled to make correct predictions due to a lack of exposure to such variations in training. For instance, consider the images in Fig. 4a and Fig. 4b, which represent the angry emotion in the edited test and training datasets, respectively.



(a) Test Image



(b) Train Image

Fig. 4: Comparison of Angry images in test vs. train

Both images in Fig.4a and Fig.4b clearly depict the angry emotion. However, when the model was trained only on one type of anger, it struggled to generalize across different variations. Additionally, the modified training set lacked nuanced angry faces, preventing the model from learning subtle features that define this emotion. Despite the increase in accuracy, the model performed better at predicting extreme versions of emotions rather than more nuanced ones. These results are reflected in Fig. 5, Fig. 6, and Fig. 7.

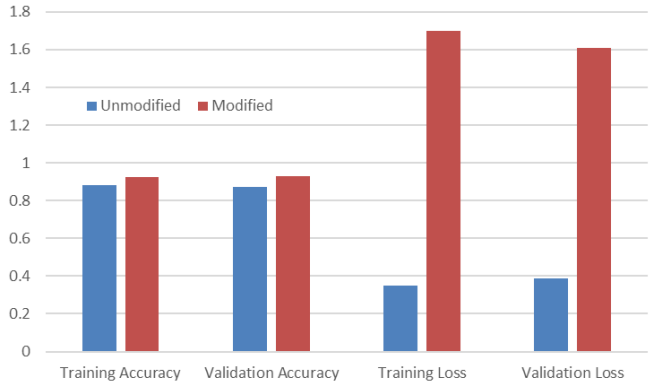


Fig. 5: Unmodified vs. Modified Happy-Neutral-Fear

Fig. 6: Unmodified vs. Modified Happy-Neutral-Angry

Fig. 7: Unmodified vs. Modified Happy-Neutral-Sad

It is also important to note that when filtering the dataset for only extreme emotions, some less extreme negative emotions were retained in the test set due to limited data availability. Removing them would have resulted in insufficient training data for effective model training, which also contributed to the increased training and validation loss.

Similar results were observed for the model trained on all five emotions—Happy, Sad, Neutral, Anger, and Fear—as shown in Fig. 8.

It is worth noting that the difference in loss (δ_{loss}) for both training and validation was smaller in this model compared to the previous six models. This is likely due to the model having

Fig. 8: Unmodified vs. Modified for All Five Emotions

access to a more diverse set of data, enabling it to make more accurate generalizations. The analysis regarding accuracy and loss follows the same pattern as with the other models.

VI. Challenges

A. Communication Challenges

In designing the communication protocol, our goal was to ensure quick and efficient data transmission. Before settling on the HTTP/HTTPS protocol, as mentioned in Section III-A, we initially implemented a low-level socket programming protocol that relied on manual serialization, deserialization, and message framing. However, this approach presented several challenges.

One significant issue arose during serialization and deserialization. We used the Python struct library, which offers two options for serializing messages: "L" (long integer) and ">I" (unsigned integer). When using the long integer, the edge device interpreted it as an 8-bit integer, while the Raspberry Pi treated it as a 4-bit integer, leading to incompatibility between the two devices. On the other hand, using the unsigned integer led to overflow issues, as the frame data from the Pi often exceeded the 4-bit limit. Additionally, the low-level socket protocol lacked built-in encryption, unlike HTTPS.

Ultimately, we chose to implement the HTTP/HTTPS protocol for several reasons. First, it abstracts away the need for manual serialization and deserialization of frames, allowing for communication using JSON, which is significantly easier to handle. Moreover, HTTPS provides inherent security features through SSL certificates, ensuring safe data transmission.

B. Model Challenges

Our initial approach involved using the Random Forest (RF) algorithm combined with CNNs for feature extraction. RF was appealing for its interpretability, ability to handle high-dimensional data, and resistance to overfitting.

As an ensemble model, RF improves accuracy by combining decision trees, mitigating the overfitting often seen in individual trees. Its interpretability was especially attractive, allowing us to identify important features in predicting emotions and facilitating model improvements. RF's ability to handle overfitting was crucial due to the extreme overfitting issues

in our initial model and the variability of human emotions. However, RF lacked a critical element for facial emotion detection: feature learning. Unlike CNNs, which learn and capture spatial data, RF relies on predefined features, making it less effective at recognizing subtle emotional nuances, leading to lower accuracy on new datasets. This limitation led us to switch to CNNs.

During emotion recognition, several issues arose. Some emotions, like sad and neutral, were difficult to distinguish, even for human observers. In real-world scenarios, a user feeling sad may display a neutral expression, leading the companion robot to misinterpret the emotion as neutral. Similarly, surprise and anger were sometimes confused, with the surprised expression of an angry person being misread.

We also experimented with multiple datasets, including AffectNet, a large dataset with nearly 1 million images. We trained on a subset of 291,651 images but found that our model performed worse with AffectNet compared to RAF-DB. This could be due to the higher variability and noise in AffectNet, which likely introduced challenges for the model. These limitations were addressed by switching over to RAF-DB because it had fewer images and thus fewer nuances within emotions for the model to learn. Additionally, the overlap in facial features in AffectNet across different emotional classes likely contributed to model confusion.

Despite AffectNet's size, its variability and potential misalignment with our use case resulted in poorer outcomes compared to RAF-DB. The higher resolution of AffectNet images may have caused the model to focus more on fine-grained details, rather than broader emotional patterns, reducing its effectiveness in emotion recognition.

VII. Conclusion and Future Work

In this study, we explored the potential of deep learning and edge computing to enhance emotion recognition capabilities in companion robots. By implementing a ResNet-50-based model within a companion robot-edge device system, we addressed the challenge of real-time emotion detection with high inference speeds, accepting a slight trade-off in accuracy to achieve this goal. Our approach utilized the RAF-DB dataset, which offered a more diverse and extensive set of images compared to traditional datasets like FER-2013 or CK+, thereby improving the model's robustness in recognizing emotions across various demographics.

The results demonstrate that real-time emotion recognition between a companion robot and an edge device is feasible, but several challenges remain. These include optimizing communication and inference speeds, maintaining the accuracy of the emotion recognition model on resource-constrained hardware, and addressing limitations of available datasets. Despite these challenges, our research highlights the significant potential of integrating emotion recognition technology into real-world applications, such as healthcare and elder care, where early detection of emotional distress could enable timely interventions and improve quality of life.

For future work, we plan several improvements. A key issue is hardware security. Although communication is secured with HTTPS, the Raspberry Pi's hardware remains physically exposed, leaving sensitive data vulnerable. Our current approach mitigates this by not saving any data on the devices, but real-time information is still accessible. Additionally, we aim to incorporate a danger detection component into the model, allowing the companion robot to assist in preventing harmful situations (e.g., a child inserting metal into an outlet, or an elderly person falling) and to contact emergency services when necessary.

Acknowledgment

This work was funded in part by the Commonwealth Cyber Initiative (CCI), an investment in the advancement of cyber R&D, innovation and workforce development (Undergraduate Student Research Program).

References

[1] T. Wang, J. Deng, X. Liu, Y. Bai, and X. Gao, "A survey of deep face recognition: Techniques, datasets, and challenges," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 10, pp. 3243–3265, 2021.

[2] R. Guo, H. Guo, L. Wang, M. Chen, D. Yang, and B. Li, "Development and application of emotion recognition technology — a systematic literature review," *BMC Psychology*, vol. 12, p. 95, 2024, accessed on August 24, 2024. [Online]. Available: https://doi.org/10.1186/s40359-024-01581-4

[3] N. Hettich, T. M. Entringer, H. Kroeger, P. Schmidt, A. N. Tibubos, E. Braehler, and M. E. Beutel, "Impact of the covid-19 pandemic on depression, anxiety, loneliness, and satisfaction in the german general population: a longitudinal analysis," *Social Psychiatry and Psychiatric Epidemiology*, vol. 57, pp. 2481–2490, 2022. [Online]. Available: https://doi.org/10.1007/s00127-022-02311-0

[4] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. IEEE, 2005, pp. 886–893. [Online]. Available: https://ieeexplore.ieee.org/document/1467360

[5] Y. Boer, L. Valencia, and S. Y. Prasetyo, "Exploring the potential of pre-trained dcnn models for facial emotion detection: A comparative analysis," in *E3S Web of Conferences*. EDP Sciences, 2023. [Online]. Available: https://doi.org/10.1051/e3sconf/202342601049

[6] J. Gong, "Challenges in transfer learning for facial expression recognition," in *Proceedings of the 2020 International Conference on ABCs*, 2020, pp. 80–88, accessed: August 24, 2024. [Online]. Available: https://users.cecs.anu.edu.au/~Tom.Gedeon/conf/ABCs2020/paper/ABCs2020_paper_v2_80.pdf

[7] M. K. Chowdary, T. N. Nguyen, and D. J. Hemanth, "Deep learning-based facial emotion recognition for human–computer interaction applications," *Neural Computing and Applications*, vol. 35, pp. 23 311–23 328, 2023. [Online]. Available: https://doi.org/10.1007/s00521-021-06012-8

[8] S. Li and W. Deng, "Reliable crowdsourcing and deep locality-preserving learning for expression recognition in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2852–2861.

[9] R. Nimmagadda, K. Arora, and M. V. Martin, "Emotion recognition models for companion robots," *The Journal of Supercomputing*, vol. 78, pp. 13 710–13 727, 2022, published online: 24 March 2022. [Online]. Available: https://doi.org/10.1007/s11227-022-04416-4

[10] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *arXiv prepring arXiv:1603.07285*, 2018, accessed on August 24, 2024. [Online]. Available: https://arxiv.org/abs/1603.07285

[11] A. Y. Ng, "Feature selection, l1 vs. l2 regularization, and rotational invariance," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 78.

[12] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 3rd ed. Upper Saddle River, NJ, USA: Pearson Prentice Hall, 2008, color Image Processing.